Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 34 (2015) 50 – 55

www.elsevier.com/locate/procedia

9th International Conference on Axiomatic Design – ICAD 2015

Robust Decision Making for Agile Systems Development Part 2: A Decomposition and Analysis

Stephen G. Barker*, Matthew P. Summers

Cranfield University, Shrivenham, Wiltshire, United Kingdom, SN6 8LA

* Corresponding author. Tel.: +44 (0) 1793 31411. *E-mail address:* s.g.barker@cranfield.ac.uk

Abstract

The need for agility in operational systems within the defence enterprise and procurement domains has been identified by many authors, and over time, there have been a number of initiatives and programmes that have sought to identify the nature of agility, and the means by which it can be defined and employed within individual cases and scenarios. These have identified impediments to the successful realization of agile practices and methods, particularly the resilience of agile decision making throughout the conceptual understanding, design and implementation of the operational system. To further investigate the extent to which this process can be implemented in a robust and reliable manner, Cranfield University created the 'Robust Enterprise-based Approach for Agility in Capability Through-life (REA^{2}CT)' framework, which provides a number of functional steps to institute a systems development lifecycle approach to producing agile solutions for use in networked systems and systems-of-systems. This paper builds upon the description of the framework [1] by applying the Axiomatic Design (AD) theory to identify where complexity exists within the requirements and design activities that underpin the framework. Using this analysis, this paper identifies 'pain points' within the REA^{2}CT framework, and suggests necessary improvements to facilitate the implementation of agility throughout the systems development lifecycle.

Peer-review under responsibility of the organizing committee of 9th International Conference on Axiomatic Design

Keywords: Systems Agility; Robust Lifecycle Approach; Axiomatic Design

1. Introduction

The customer needs (CNs) for an organisation to be capable of facilitating operational agility were discussed in an earlier paper by the same authors [1], and can be seen as follows:

[Initial CNs]

1. Identification of the rapidity and nature of response required
2. Develop organizational systems/services to facilitate creation of an operational architecture that promotes agility
3. Create initial architecture of configuration(s) to facilitate agile decision making
4. Define/validate potential operational scenarios/configurations

[Subsequent CNs]

5. Choose appropriate configuration to suite operational need
6. Ability to respond to requirement change rapidly

Broadly, these can be seen to map to the issues identified earlier [1]. Having identified needs for facilitating operational agility, the REA^{2}CT framework will now be examined.

2. REA^{2}CT framework

As described [1], the REA^{2}CT Framework is a means to facilitate understanding of the operational agility need, and to model and institute an enterprise-wide structure to address the meeting of that need. The key stages of REA^{2}CT are:

1. Construct the Time Dependency Matrix
2. Identify necessary Reconfiguration Activities

Peer-review under responsibility of the organizing committee of 9th International Conference on Axiomatic Design
doi:10.1016/j.procir.2015.07.003

3 Assess the Enterprise Agility Profile
4 Confirm the Operational Agility Level
5 Institute the System of Systems Process
6 Adapt the Lifecycle Management Process

Having identified the key stages of the REA^{2}CT Framework, it is necessary to analyze it to ensure conformance with requirements described earlier. The technique chosen for this analysis was Axiomatic Design [2, 3], and this will be briefly introduced in the following section.

3. Axiomatic Design

A key element of engineering is the process of design, which encompasses both synthesis and analysis [3]. Design necessarily must encompass the human element, but in order to govern 'good' design practice, the synthesis process needs a set of scientific principles [3]. This is embodied by Axiomatic Design (AD) theory [2, 3], which has seen widespread use in a number of fields, including healthcare [4], safety [5], and human factors [6]. AD theory identifies a set of axioms, domains, and hierarchies by which design can be structured. The two axioms are those of:

- Independence
- Information

Axiom One requires that Functional Requirements (FRs – which define functions) be independent of one another, which enables each FR to be satisfied *without* affecting any of the other FRs. Thus there is no coupling of FRs where it can be avoided, and the design remains as uncomplicated as possible. Axiom Two provides a quantitative measure of the merits of a given design, and thus it is useful in selecting the best among the designs which satisfy axiom one [3]. As Guenov and Barker [7] point out, generally, the design which uses the least information is superior. This analysis will concentrate upon axiom one.

The four domains of design [2] are: Customer Needs (CN), Functional Requirements (FR), Design Parameters (DP). This analysis will focus on the process of embodying design parameters from functional requirements. This process is shown in figure 1:

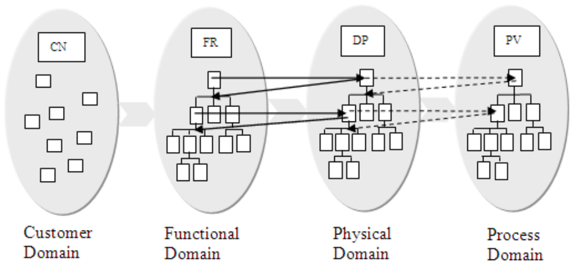

Fig 1. Domains of Axiomatic Design [7]

The relationship between FRs and DPs can be expressed as:

$$\{FR\} = [A]\{DP\} \quad (1)$$

Where [A] is called the *design matrix* [3]. Three types of design can exist [3], as follows:

- *Uncoupled*, where each of the FRs can be satisfied by a single DP, and the design matrix is diagonal
- *Decoupled*, where the design matrix is triangular, and the independence of FRs can be satisfied should the DPs be determined in a proper sequence
- *Coupled*, any other form of design, when what is termed a 'full matrix' occurs

An uncoupled design represents a good logical solution, and a decoupled design a solution that can be viable should the DPs be determined in a way that guarantees the independence of FRs. Where the design is coupled, however, an iterative process of analysis and intervention must be followed to render the design at least decoupled.

Axiomatic Design includes a number of theorems [2] to guide and identify design states, the most relevant to this work being:

- Theorem One (Coupling due to Insufficient number of DPs), when the number of DPs is less than the number of FRs, either a coupled design results, or the design cannot be satisfied
- Theorem Four (Ideal Design) where the number of DPs is equal to the number of FRs and the FRs are always maintained independent from each other

The analysis of the REA^{2}CT Framework in the following section applies Axiomatic Design theory to understand the extent to which the key stages or Design Parameters of the REA^{2}CT framework achieve the requirements for agility.

4. Analysis of REA^{2}CT framework

The relationship between CNs for agility, and FRs for the REA^{2}CT framework has previously been identified [1], and broken down into initial Functional Requirements (FRs) and associated Design Parameters (DPs). This is shown in table 1.

Table 1. Initial Functional Requirements and Design Parameters

Functional Requirements (FR)	Design Parameters (DP)
1. Identify Effect/Need	1. Time Dependency Matrix
2. Define required services	2. Reconfiguration activities
3. Create Service Oriented Architecture (SoA)	3. Enterprise Agility Profile
4. Identify and define possible configurations	4. Operational Agility Level
5. Choose/Deploy appropriate configuration	5. System-of-Systems (SoS) process
6. Reconfigure	6. Lifecycle Management Process

The Axiomatic Design process was then applied to establish the independence of FRs, thus satisfying the Independence Axiom. For the sake of clarity, and to expose the stages by which the research was conducted, the following analysis is split into sub-sections. The process by which the research was conducted will be reflected upon more fully once the analysis is described.

4.1. Step 1: Initial derivation of Design Matrix

As the design parameters (DPs) affect the achievement of functional requirements (FRs), the relationships between each of the FRs and DPs were identified and mapped onto a design matrix, with an identified relationship between the terms denoted by an X, as shown in figure 2.

FR\DP	1	2	3	4	5	6
1	X					X
2	X				X	
3	X	X	X		X	X
4	X			X	X	X
5	X				X	X
6						X

Fig 2. Initial Design Matrix for REA^{2}CT

4.2. Step 2: Revision upon Reflection

No further analysis or consideration to this matrix was given for a week as the authors were engaged on other tasks. Reconvening after the week had passed; the authors reviewed the matrix and reflected on their initial decisions and analysis. This prompted a number of changes to the initial matrix as there were several aspects of the original analysis and decisions taken that the authors, on reflection, disagreed with. This resulted in the matrix shown in figure 3.

FR\DP	1	2	3	4	5	6
1	X					
2	X	X				
3		X	X			X
4	X		X	X	X	X
5	X				X	X
6				X		X

Fig. 3. Initial Design Matrix for REA^{2}CT (after reflection)

A significant change was the renaming of FR4 from "Identify and define possible configurations" to "Consider all possible configurations". This was done to more accurately reflect that this is what is required at that point.

Further changes involved the deleting or adding of relationships; these changes, and the rationale behind them, being described in table 2.

Table 2. Rational behind changes to the Design Matrix in Step 2 of the analysis

6-1 (DP6-FR1)	**Deleted** – any Lifecycle Management Process can be applied, and done so retrospectively.
1-2	Becomes inferred relationship, as service has to deliver against time sensitivity.
2-2	**Added** – Services underpin reconfiguration activities.
5-2	**Deleted** – process has no direct and explicit link to individual service(s).
1-3	**Deleted** – SoA irrelevant from time perspective because services are already defined with respect to time (SoA merely arranges the configuration).
5-3	**Deleted** – SoA can be created without consideration of the SoS process that will deliver the configurations.
3-4	**Added** – Enterprise Agility Profile has a bearing on ability to instantiate possible configurations.
4-6	**Added** – Reconfigure requires Operational Agility.

4.3. Step 3: Re-ordering of DPs to reduce coupling

At this point, whilst coupling within the design matrix had been reduced, the matrix was still coupled. As such, the design parameters were reordered to reduce the exhibited coupling further.

FR\DP	1	2	3	6	4	5
1	X					
2	X	X				
3		X	X	X		
4	X		X	X	X	X
5	X			X		X
6				X	X	

Fig. 4. REA^{2}CT Design Matrix after first re-ordering of DPs

This process involved a number of changes to the Design Matrix. Firstly, DP6 (Lifecycle Management Processes) was moved to sit between DP3 and DP4 because Lifecycle Management Processes can be defined prior to Operational Agility. This is shown in figure 4. Upon further analysis of the logic chain of implementing the DPs, moving DP6 meant that DP5 (SoS Process) also had to be reordered as system-of-systems profiling had to occur before the implementation of Lifecycle Management Processes, given it is a driver of this process. This resulted in the following design matrix figure 5.

FR\DP	1	2	3	5	6	4
1	X					
2	X	X				
3		X	X			
4	X		X	X	X	X
5	X			X	X	
6						X

Fig. 5. REA^{2}CT Design Matrix after second re-ordering of DPs

Following the DP reordering, a re-analysis of the FR-DP relationships was also undertaken. This resulted in a further two relationships being deleted, as described in table 3.

Table 3. Rational behind changes to the Design Matrix in Step 3 of the analysis

6-3 (DP6-FR3)	**Deleted** – Service Oriented Architecture can be defined independently of Lifecycle Management Process.
6-6	**Deleted** – Reconfiguration now mapped against Operational Agility, with Lifecycle Management Processes mapped against possible configurations (FR5).

At this point of the analysis, only one of the functional requirements, FR4, had been changed and Design Parameters 5 and 6 reordered. Therefore the FR-DP breakdown prior to re-examining the Functional Requirements was as described in table 4:

Table 4. Functional Requirements and Design Parameters after step 3 of the analysis

Functional Requirements (FR)	Design Parameters (DP)
1. Identify Effect/Need	1. Time Dependency Matrix
2. Define required services	2. Reconfiguration activities
3. Create Service Oriented Architecture (SoA)	3. Enterprise Agility Profile
4. Consider all possible configurations	5. System-of-Systems (SoS) process
5. Choose/Deploy appropriate configuration	6. Lifecycle Management Process
6. Reconfigure	4. Operational Agility Level

As with the results of the previous analysis steps, this further reduced the coupling exhibited within the matrix, but still left a coupled matrix. Consequently, the functional requirements were then re-examined.

4.4. Step 4: Re-examining the FRs

Examination of the functional requirements showed that FR4 (Consider all possible configurations) could be considered a composite requirement and thus be disaggregated. Therefore, it was decided to split this into two functional requirements: FR4a – Identify Possible Configurations against the SoS, and FR4b – Institute Possible Configurations. These derived functional requirements were then examined against the design parameters and relationships determined. FR4a could be undertaken independently of Operational Agility, which is now facilitated by possible configurations, rather than being a factor in affecting them. The overall System of Systems (SoS) would need to be examined as part of this requirement to 'harmonise' individual elements of the SoA.

It was determined that for FR4b (institute possible configurations), in addition to existing FR-DP relationships, the user must consider the following design parameter (DPs):

- use of reconfiguration activities;
- baseline against the Enterprise Agility Profile (EAP);
- Determine the required SoS process; and
- link to the Lifecycle Management Plan (which facilitates FR4b).

Furthermore, in light of the changes to FR4, FR5 was rewritten to be more meaningful as "Select Required Configuration" without compromising the FR-DP relationships. The revised FRs and DPs are shown in table 5.

Table 5. Revised FRs and DPs after step 4 of the analysis

Functional Requirements (FR)	Design Parameters (DP)
1. Identify Effect/Need	1. Time Dependency Matrix
2. Define required services	2. Reconfiguration activities
3. Create Service Oriented Architecture (SoA)	3. Enterprise Agility Profile
4a. Identify possible configurations against SoS	5. System-of-Systems (SoS) process
4b. Institute possible configurations	6. Lifecycle Management Process
5. Select Required Configuration	4. Operational Agility Level
6. Reconfigure	

This generated a further iteration of the design matrix, shown in figure 6:

FR\DP	1	2	3	5	6	4
1	X					
2	X	X				
3		X	X			
4a	X		X	X		
4b	X	X	X	X	X	
5	X			X	X	
6						X

Fig. 6. Revised matrix after step 4 of the analysis

4.5. Step 5: Re-defining the DPs

As the matrix at figure 6 shows, the key issue at this point was that the analysis decisions produced an off-square matrix, causing a coupled design as described by theorem one of Axiomatic Design [3]. This indicated that REA^{2}CT required modification to meet the re-defined functional requirements.

This modification required a further iteration of analyzing and revising the design parameters. Again, the authors felt that one of the parameters could be classed as composite and broken into two more specific ones. DP4 (Operational Agility Level) was split into two better focused design parameters: DP4a – Operational Agility Profile, and DP4b – Operational Doctrine. Whilst both were contributors to operational agility, we had now split the "what would be implemented" (DP4a) and the "how it would be used operationally" (DP4b) aspects. This precipitated analysis of the relationship of the two new design parameters against the seven functional requirements. As a result, three relationships were identified in table 6.

Table 6. Revised FRs and DPs after step 4 of the analysis

4a-5 (DP4a-FR5)	**Added** – Required configuration depends on Operational Agility for viability.
4a-6	**Added** – Ability to effectively reconfigure depends on Operational Agility Profile.
4b-6	**Added** – Operational Doctrine governs Reconfiguration options and possibilities.

These relationships were then added to the design matrix, which is shown in figure 7:

FR\DP	1	2	3	5	6	4a	4b
1	X						
2	X	X					
3		X	X				
4a	X		X	X			
4b	X	X	X	X	X		
5	X			X	X	X	
6						X	X

Fig. 7. Design Matrix showing re-defined DPs

This produced a square matrix, which was decoupled. As such, this can facilitate independent realization of the functional requirements, provided that the design parameters are actioned in the particular sequence identified in figure 7.

4.6. The final FR-DP relationship mapping

The out of sequence numbering was renumbered 1-7 for both functional requirements and design parameters to generate the final analysis output shown at table 7.

Table 7. Final list of FRs and DPs

Functional Requirements (FR)	Design Parameters (DP)
1. Identify Effect/Need	1. Time Dependency Matrix
2. Define required services	2. Reconfiguration activities
3. Create Service Oriented Architecture (SoA)	3. Enterprise Agility Profile
4. Identify possible configurations against SoS	4. System-of-Systems (SoS) process
5. Institute possible configurations	5. Lifecycle Management Process
6. Select Required Configuration	6. Operational Agility Profile
7. Reconfigure	7. Operational Doctrine

The final design matrix is therefore that which appears at figure 8.

FR\DP	1	2	3	4	5	6	7
1	X						
2	X	X					
3		X	X				
4	X		X	X			
5	X	X	X	X	X		
6	X			X	X	X	
7						X	X

Fig 8. Final Design Matrix

5. Reflection and conclusions

The initial design matrix (figure 2) was created from the FR-DP listing in table 2, and revealed a square, but coupled matrix. As described in section 5.2, other work then intruded to mean that further work on the analysis was delayed by a week. Upon returning to the analysis, and in order to re-familiarize the authors with the task at hand, the process of deriving the initial design matrix was repeated, resulting in figure 3.

That this produced a different matrix, still coupled but less so than before, is extremely interesting. Many authors, for example Gibbs [8] and Fry et al [9] emphasize the importance of reflection to improve practice, and although no further work had been done on the research, a number of informal discussions had taken place between the authors, indicating that they had, at least sub-consciously, been considering the issue, and thinking the process through. Given the difference in the matrix in figure 3 (from figure 2), the process of reflection clearly influenced subsequent analysis and understanding. It would be of some interest to understand what outcomes would have been achieved had the initial design matrix been retained as the basis for analysis, and indeed what understanding would then have been predicated upon those outcomes.

Faced with a coupled matrix, the analysis proceeded to apply the Axiomatic Design approach [3] of re-ordering the design parameters [DPs] to see if a change in order might satisfy the functional requirements [FRs]. Although this proved not to be the case, it enabled a clearer understanding of FR-DP relationships, and caused a further reduction in design coupling. This led to a re-examination of FRs, resulting in a partial decomposition of the FRs, as described in figure 6.

This caused theorem one of axiomatic design to apply to the research, and caused a re-definition and partial decomposition of DPs to be effected, the outcome being a once-more square matrix as depicted in figure 7.

The premise of evaluating the rigour of the REA^{2}CT Framework stemmed from a renewed interest in facilitating the process of agile decision making, and the subsequent need to ensure that REA^{2}CT robustly met the requirements of agile systems procurement and operation. The choice of Axiomatic Design to facilitate this evaluation came from one of the authors' previous experience with the technique [4], and on completion of the analysis, both authors felt that the process – coupled to reflective practice - had led to positive change within REA^{2}CT, leading to a clearer, more logical framework that better met its requirements.

6. Future Work

The analysis of REA^{2}CT to date has been mainly at a high level, and the framework may well benefit from a more in-depth study and decomposition of the FR-DP hierarchy to further clarify the relationship between functional requirements and design parameters. To this end, an iterative process of applying REA^{2}CT to a case study, followed by reflection and renewed application of the Axiomatic Design theory, may lead to an interesting detailed decomposition that would yield further benefits for the REA^{2}CT framework.

Given the earlier reflection that the reflective period leading to revision of the initial design matrix was critical to the outcomes that were achieved, it would also prove extremely interesting to re-undertake the analysis using the original, initial design matrix described in figure 3, and to evaluate the likely difference in outcomes between the two analyses. Given increasingly ever higher workloads, it would be interesting to consider the extent to which reflective practice in design might alleviate wrongful design decision making, and the extent to which Axiomatic Design might facilitate this. A potential title for such research might be: "Time critical influence on outcomes from the Axiomatic Design process of analysis and evaluation".

Acknowledgements

The authors would like to acknowledge the contribution and significant involvement of their colleagues Tim Mackley, Phil John, and John Deane in the original research and development of the REA^{2}CT Framework.

References

[1] Barker SG, Summers MP. Robust Decion Making for Agile Systems Development: Exploring the Paradigm. ICAD 2015 conference proceedings. Florence. Italy. September 2015.
[2] Suh NP. The Principles of Design. Oxford University Press US. 1990
[3] Suh NP. Axiomatic Design: Advances and Applications. Oxford University Press US. 2001.
[4] Farid AM, Khayal IS. Axiomatic Design Based Volatility Assessment of the Abu Dhabi Healthcare Labor Market (parts one and two). ICAD 2013 conference procedings. Worcester, Mass, US. June 2013.
[5] Sadeghi L, Mathieu L, Tricot N, Al-Bassit, L. Toward Design for Safety (parts one and two). ICAD 2013 conference procedings. Worcester, Mass, US. June 2013.
[6] Ghemraoui R, Mathieu L, Tricot N. Systematic Human-Safety Approach Based on Axiomatic Design Principles. ICAD 2009 conference procedings. Campus de Caparica. March 2009.
[7] Guenov MD, Barker SG. Application of Axiomatic Design and Design Structure Matrix to the Decomposition of Engineering Systems. Sys. Eng 2005; 8(1).
[8] Gibbs G. Learning by doing: A guide to teaching and learning methods. Oxford Centre for Staff and Learning Development, Oxford Polytechnic. London: Further Education Unit. ISBN 1-85338-071-7
[9] Fry H, Ketteridge S, and Marshall S (eds.). Teaching and Learning in Higher Education: Enhancing Academic Practice. 3rd ed. 2009. Routledge, UK